공학 연구논문집 제28권 제2호 pp. 81~92, 울산대학교.
Journal of Engineering Research Vol. 28, No. 2, pp. 81~92, 1997
University of Ulsan.

확장 페트리 네트의 시간 개념에 관한 조사

B. Dorjsuren · 김규년
컴퓨터 정보통신 공학부

<요 약>

본 논문에서 우리는 실시간 동시발생 시스템을 설계 및 분석할 때 고려되는 시간 개념과 시간적인 제한(timing constraint)들의 역할에 대해 알아보고, 시간 관련 페트리 네트 확장들을 자세히 살펴봐서 시간 개념이 어떤 방향으로 응용되어 가는가를 조사한 것을 소개한다. 확장된 페트리 네트들은 각각 자신의 방법으로 일반 페트리 네트에 시간 개념을 추가시켰다. 시간 개념이 주로 트랜지션 발사 허용 및 점화 규칙에 추가되었기 때문에 분류 기준을 발사 허용 및 점화 규칙의 관점과 시간적인 제한의 관점으로 정하였다. 조사 결과 실시간 동시발생 시스템의 설계와 성능 분석을 가장 잘 도울 수 있는, 요구 사항을 모형화 하는데 제일 적합한 페트리 네트를 찾고자 한다.

키워드: 실시간, 동시발생 시스템, 시간, 시간 제한, 페트리 네트.

A Survey on the Time Concepts of Extended Petri Nets

Dorjsuren Baramsai · Kyoo-Nyun Kim
School of Computer Engineering and Information Technology

<Abstract>

In this paper, we will find out about the concepts of time and timing constraints and their roles in designing and analyzing of the real-time concurrent systems, and review the time related extensions of Petri net for a survey on their implementation direction of time concept. Extended Petri nets are added the concept of time into the original Petri net using their own methods. Time concepts are mostly added to the enabling

and firing rules, so therefore, we classify Petri nets from the enabling and firing rules point of view and the timing constraints point of view. As a result we hope to find the Petri net that can help design and analysis of concurrent real-time system, and that is mostly suitable for modeling the requirements specification.

Keyword: Real-time, concurrent systems, time, timing constraints, Petri nets.

1. 서 론

실시간 시스템은 외부 자극(stimuli)에 때에 알맞은 반응(timely response)을 하는 특색을 이룬다[3]. 자극에 대한 알맞은 반응을 얻기 위해 시스템에 부과된 시간 관련 제한들이 있으며 이들 시간 제한들의 고려 하에서 시스템이 실행을 해야 한다.

보통, 실시간 시스템이 태스크들의 집합으로 모델링 된다. 각 태스크는 두 개의 사건에 의해 특징이 묘사되며, 그 중의 하나가 태스크의 실행이 시작하는 시작 사건이고 다른 하나는 태스크가 실행을 마치는 종료 사건이다. 태스크는 두 사건 사이에서 실행을 한다.

페트리 네트는 동시 발생의 시스템을 모델링하고 분석할 수 있는 능력 때문에 최근 들어 많은 인기를 얻었다. 하지만, 페트리 네트에는 시간 개념이 명백히 제공되지 않아, 실시간 시스템에서의 사용을 제한하고 있다. 따라서, 페트리 네트를 시간적 행동(behavior) 분석[7][8][9][10]과 성능 평가[11][12]의 범위에서 확장하는 노력들이 많이 나왔으며, 여기서 우리는 이러한 페트리 네트 확장들을 비교하고자 한다. 대부분의 확장들은 원래의 페트리 네트의 발사 허용(enabling)과 점화(firing) 규칙에 시간적 제한을 부과시켜서 만들어졌다. 그래서, 우리는 비교를 발사 허용과 점화 규칙의 관점과 시간적인 제한의 관점에 맞추겠다.

1. 페트리 네트의 개요

1.1 페트리 네트 정의와 표기법

페트리 네트[1](Petri net)는 1962년 독일 Bonn대학의 C.A.Petri의 박사학위 논문으로서, 비동기적이고 동시발생 시스템에 있어서의 정보 흐름을 모델링 하는 그래프로 처음 개발되었다. 페트리 네트에서 이벤트(event)를 트랜지션(transition)이라고 한다[2]. 트랜지션이 발사하기 위해 몇 가지 조건(condition)이 만족해야 할 때도 있을 것이며, 이들 조건들에 관한 정보는 플레이스(place)에 포함된다. 어떤 플레이스들은 트랜지션의 입력(input)으로 보여질 수 있으며, 이 트랜지션이 발사하기 위해 필요한 조건에 해당된다. 다른 플레이스들은 트랜지션의 출력(output)이 될 것이고, 트랜지션이 발사함으로서 영향을 받은 조건들에 해당이 된다. 트랜지션, 플레이스, 그리고 그들 사이의 관계는 페트리 네트의 기본 구성요소들이다.

[정의 1.1] 페트리 네트 $PN = (P, T, A, w)$는 4-tuple이다. 여기서,

P - 플레이스들의 한정된 집합,

T - 트랜지션들의 한정된 집합,

A - 호선들의 집합이며, 집합 $(P\times T) \cup (T\times P)$의 부분 집합이고,
w - 가중치(weight) 함수 $w : A \rightarrow \{ 1,2,3,\ldots \}$이다.

호선은 페트리 네트의 그래픽 표현에서 비롯된 것이다. 이 그래픽 표현에는 두 가지 종류의 노드가 있으며 플레이스가 원으로, 트랜지션이 막대기(bar)로 표시된다. 그림 1.1에서 보여주는 것과 같이 페트리 네트는 두 노드를 여러 개의 호선으로 연결하는 것을 허용하며 그 개수를 호선의 가중치(weight)로서 표시한다.

$P = \{p_1, p_2\},\ T = \{t_1\},\ A = \{(p_1, t_1), (t_1, p_2)\},$

$\omega\ (p_1, t_1) = 2,\ \omega\ (t_1, p_2) = 1$

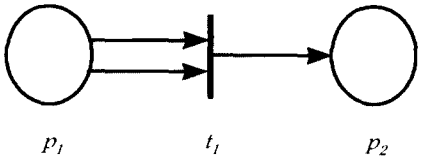


그림 1.1

그리고, 각 플레이스와 트랜지션의 대해 입력(input) 집합 I과 출력 (output) 집합 O를 다음과 같이 정의한다.

$$I(t_j) = \{p_i : (p_i, t_j) \in A\},\quad O(t_j) = \{p_i : (t_i, p_j) \in A\},$$
$$I(p_j) = \{t_i : (t_i, p_j) \in A\},\quad O(p_j) = \{t_i : (p_j, t_i) \in A\}$$

1.2 페트리 네트 마킹과 상태 공간

페트리 네트 그래프에서 특정 조건들이 만족되는 상황을 표시하기 위해 플레이스에 토큰(token)을 할당한다.

[정의 1.2] 페트리 네트의 마킹(marking) x는 함수 $x : P \rightarrow N$이다.

그럼으로, 페트리 네트의 플레이스의 수가 n이라면 마킹은 벡타 $\mathbf{x} = [x(p_1), x(p_2), \ldots, x(p_n)]$를 정의한다. 벡타의 i번째 항이 플레이스 p_i의 토큰 수를 나타내는 것이다. 그래프 상에 토큰은 해당되는 플레이스에 위치한 검은 점으로 표시된다.

[정의 1.3] 페트리 네트 (P, T, A, w)와 초기 마킹 x_0에 대해 5-tuple (P, T, A, w, x_0)를 마킹된 페트리 네트라고 한다.

일반적으로, 마킹된 페트리 네트를 그냥 페트리 네트라고 부른다.

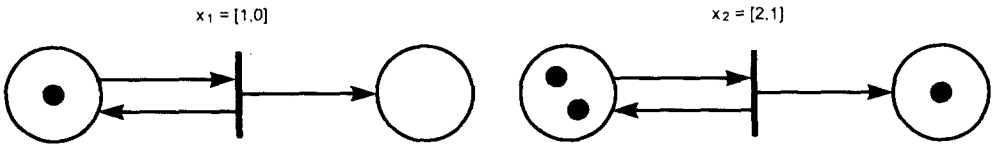


그림 1.2

[정의 1.4] 마킹 $\mathbf{x}=[x(p_1), x(p_2), \ldots, x(p_n)]$를 페트리 네트의 상태(state)라고 한다. 그럼으로, 상태 공간(state space) X이 각 항이 음이 아닌 정수 마킹인 모든 n-차원 벡타들로 정의된다. 즉,
$X=\{[y_1, y_2, \ldots y_n] \mid y_i \in N,\ i=1,\ldots n,\}$ 이다.

트랜지션(또는, 사건)이 발사 허용되기 위해서 각 입력 플레이스들에 토큰이 나타나야 한다.

[정의 1.5] 페트리 네트의 트랜지션 $t_j \in T$가 다음 조건을 만족하면 발사 허용된다(enabled)고 한다.
모든 $p_i \in I(t_j)$에 대해 $x(p_i) \geq w(p_i, t_j)$

1.3 페트리 네트의 실행

페트리 네트는 트랜지션을 발사함(firing)으로서 실행된다. 트랜지션은 자신의 입력 플레이스들에서 토큰을 제거하고 출력 플레이스에 새로운 토큰을 생성함으로서 발사한(fire)다 (그림 1.2).
발사 허용된 트랜지션이 발사할 수 있다. 트랜지션이 발사함으로서 페트리 네트의 상태가 전환되며, 이를 위한 전환 장치(mechanism)가 정의되어 있다.

[정의 1.6] 트랜지션 t_j 발사 허용되어질 때마다 발사할 수 있다. 트랜지션이 발사한 결과로 다음과 같이 정의된 새로운 마킹 상태 x'가 나온다.

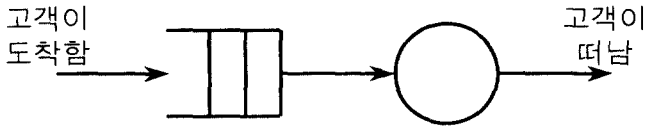
$$x' = x(p_i) - w(p_i, t_j) + w(t_j, p_i) \qquad i = 1, \ldots, n$$

이와 같은 구조를 사용하여 여러 가지 많은 시스템들을 간단하고, 이해하기 쉽게 구현할 수 있다. 예를 들어, 간단한 큐잉(queueing) 시스템을 그림 1.3과 같이 모델링 할 수 있다.

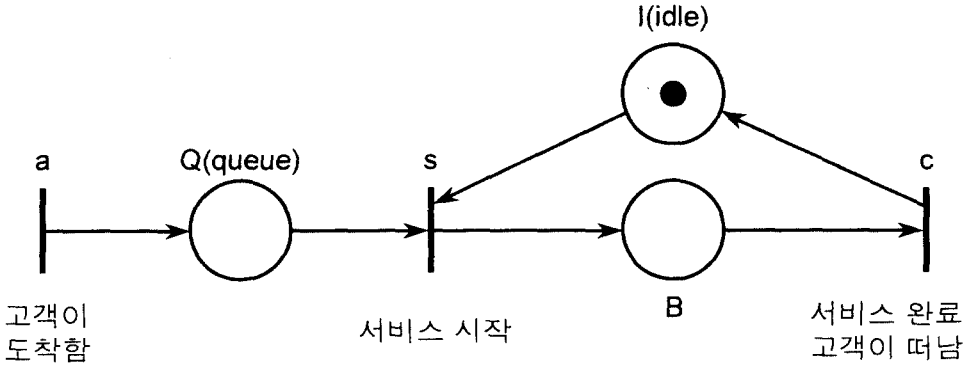
2. 실시간 시스템의 시간 개념

실시간 시스템은 외부 자극(stimuli)에 대해 알맞은 반응(timely response)을 하는 특색

을 이룬다[3]. 일반적으로, 자극에 대한 반응은 연속된 태스크 실행으로 이루어지며, 태스크 실행은 보통 그의 시작 시간, 실행 시간, 그리고 최종 기한(deadline)으로 특성이 묘사된다. 실시간 시스템에서 즉각적으로 반응을 얻기 위해 태스크 실행 스케이듈을 제시하여, 설계하고, 부과된 시간적 제한들의 고려 하에서 실행이 되어야 한다. 스케이듈이 시간적인 제한들을 만족한다는 것을 입증하는 과정을 스케이듈 가능성 분석(schedulability analysis)



간단한 큐잉(queueing) 시스템



[그림 1.3] 큐잉 시스템의 페트리 네트 모델

이라고 한다[4][5]. 스케이듈 가능성 분석을 위한 기술들은 주로 설계나 실행의 관점에서 고려되어졌다. Stoyenko등이 언어와는 독립적인 스케이듈 가능성 분석 기술들[5]을 제시하였는데, 프로그램의 구조와 응용 및 하드웨어 설정에 관한 정보들을 이용해서 실시간 시스템이 시간적인 제한들을 만족하는지 검증한다. Haban과 Shin등이 실시간 모니터링 방법[4]을 써서 모니터한 자료들을 분석하고, 분석된 결과를 호스트 운영체제로 되돌려 모니터하는 태스크들의 동적 스케이듈링 및 검증에 공급하는 접근 방법을 사용하고 있다.

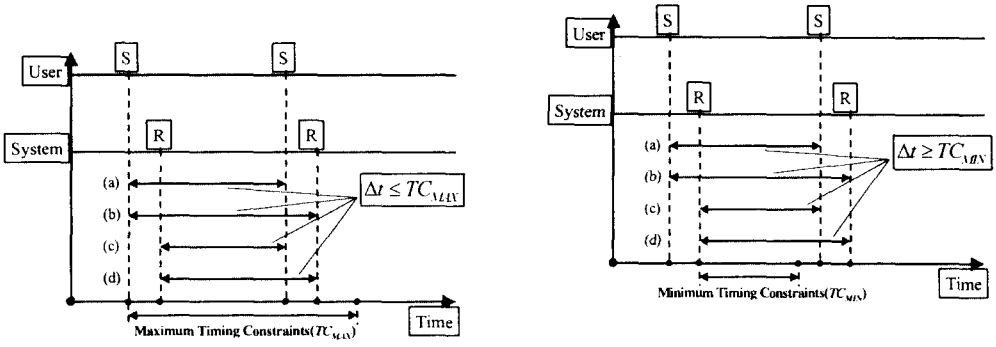
보통, 실시간 시스템이 태스크들의 집합으로 모델링 된다. 각 태스크는 두 개의 사건에 의해 특징이 묘사되며, 그 중의 하나가 태스크의 실행이 시작하는 시작 사건이고 다른 하나는 태스크가 실행을 마치는 종료 사건이다. 태스크는 두 사건 사이에서 실행을 한다. 그에 추가적으로, 실시간 시스템의 태스크들은 [6]에서 정의한 바와 같은 시간적인 제한들을 갖는다. Dasarathy는 시간적인 제한들을 여러 가지로 분류한 것 중에 다음과 같은 세 가

지 제한이 있다: 1) 최대 시간 제한은 두 사건 사이에서 경과될 수 있는 최대의 시간이고; 2)최소 시간 제한은 두 사건 사이에서 경과되어야 하는 최저의 시간이며; 3) 지속 시간 제한은 태스크가 지속할 수 있는 최대의 시간이다.

1) 최대 시간 제한은 다음과 같은 4가지 종류가 있다;
   A. 자극 대 자극 조합(S-S): 최대 시간이 두 자극 사이의 간격에 주어진다.
      예) 전화하는 사람이 첫 숫자를 눌은 후, 두 번째 숫자를 20초 안에 눌러야 한다.
   B. 자극 대 응답 조합(S-R): 최대 시간이 자극과 응답 사이의 간격에 주어진다.
      예) 전화하는 사람이 수화기를 들어올린 후 2초안에 신호음을 받아야 한다.
   C. 응답 대 자극 조합(R-S): 최대 시간이 응답과 자극 사이의 간격에 주어진다.
      예) 신호음을 받은 후, 전화하는 사람이 첫 숫자를 30초안에 눌러야 한다.
   D. 응답 대 응답 조합(R-R): 최대 시간이 두 응답 사이의 간격에 주어진다.
      예) 연결이 이루어진 뒤, 호출 당하는 사람이 벨이 울리는 소리를 수신한 후 0.5초 이내에 전화하는 사람이 되돌아오는 벨 소리를 수신해야 한다.

2) 최소 시간 제한의 4가지 종류는 다음과 같다;
   A. 자극 대 자극 조합(S-S): 최소 시간이 두 자극 사이의 간격에 주어진다.
      예) 전화할 때 두 숫자를 누르는 데 최소한 0.5초가 경과해야 한다.
   B. 자극 대 응답 조합(S-R): 최소 시간이 자극과 응답 사이의 간격에 주어진다.
      예) 전화하는 사람이 0을 누른 후 시스템은 응답하기 전에 15초 동안 기다린다.(아마도, 이것은 송신자가 교환원을 통하는 호출 과정을 끝마칠 수 있게 하기 위한 것이다.)
   C. 응답 대 자극 조합(R-S): 최소 시간이 시스템의 응답과 다음 자극 사이의 간격에 주어진다.
      예) 어떤 응용 프로그램에서, 시스템이 여러 개의 포트로부터 오는 요청을 처리하느라 바쁠 수도 있다. 따라서, 특정 포트로부터 그의 마지막 요청 이후의 자극을 시스템이 받아들이기 위해서 일정한 시간이 지나야 한다.
   D. 응답 대 응답 조합(R-R): 최소 시간이 두 응답 사이의 간격에 주어진다.
      예) 연결이 이루어진 뒤, 호출 당하는 사람이 벨이 울리는 소리를 들은 후 0.5초 이내에 전화하는 사람이 되돌아오는 벨 소리를 들을 수 있어야 한다.

이것을 그림으로 설명하면 다음과 같다.



[그림 2.1] 최대 및 최소 시간 제한의 종류들

3. 페트리 네트의 시간 관련 확장들

3.1 발사 허용과 점화 규칙의 관점

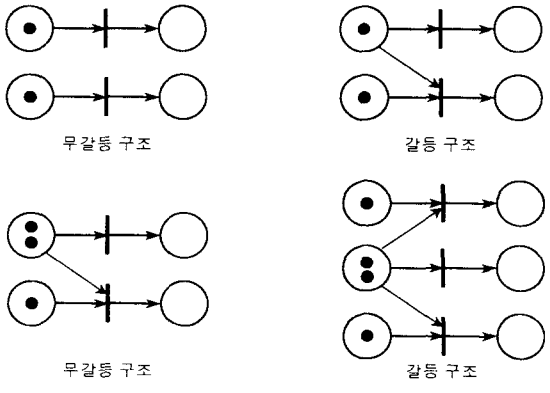
발사 허용 규칙을 다음과 같이 두 종류로 분류한다: 타입이 없는(typeless) 발사 허용 규칙과 타입이 있는(typed) 발사 허용 규칙이 이것이다. 타입이 없는 규칙은 토큰을 모두 같은 것으로 취급한다. 따라서, 트랜지션 t_j가 발사 허용되기 위해 t_j의 각 입력 플레이스에 토큰이 나타났는지 만 확인하면 된다. 반대로, 타입이 있는 발사 허용 규칙에서는 각 토큰이 서로 다른 속성을 가질 수 있게 토큰들을 개별적으로 취급한다. 그러므로, 트랜지션 t_j가 발사 허용되기 위해 t_j의 각 입력 플레이스에 토큰이 나타났는지를 확인하는 것 이외에 토큰의 타입이 알맞은 조화를 이루는지 확인해야 한다.

이와 같은 발사 허용 규칙들에 기반을 두어 점화 규칙도 타입 없는 점화 규칙과 타입 있는 점화 규칙으로 분류할 수 있다. 타입이 없는 점화 규칙의 경우 호선의 가중치에 의존하여 트랜지션의 각 입력 플레이스에서 토큰이 제거되고, 각 출력 플레이스에 추가된다. 일반 페트리 네트가 타입이 없는 발사 허용 규칙과 점화 규칙을 사용한다. 이에 반면에, 타입이 있는 점화 규칙을 따르는 네트에 경우 타입 있는 트랜지션을 점화하는 것은 트랜지션의 각 입력 플레이스에서 특정 타입의 토큰들을 제거하고, 각 출력 플레이스에 특정 타입의 토큰들을 첨가한다. 그렇게 하기 위해, 어떤 타입의 토큰들의 조합이 트랜지션을 발사 허용할 수 있는 지와, 어떤 타입의 토큰들의 조합이 입력 플레이스들에서 제거되고 출력 플레이스에 추가되어야 할 것인지를 알 수 있는 테이블(table)이 사용된다. 채색 페트리 네트(Colored Petri net)가 타입 있는 발사 허용과 점화 규칙을 둘 다 사용한다.

논의를 쉽게 하기 위해, 발사 허용된 트랜지션이 얼마나 빨리 점화하는가에 기반을 두어 트랜지션 점화 모드를 약한 점화 모드(weak firing mode)와 강한 점화 모드(strong firing mode)로 분류할 수 있다. 약한 점화 모드는 발사 허용된 트랜지션이 반드시 점화하도록

강요하지 않는다. 다시 말해서, 발사 허용된 트랜지션은 점화할 수도 있고 안할 수도 있다. 일반 페트리 네트가 이 모드를 사용한다. 강한 점화 모드는 발사 허용된 트랜지션이 곧 점화하기를 강요한다. 트랜지션이 발사 허용된 즉시 점화하기를 강요하는 것이다. 강한 점화 모드가 무갈등(conflict-free) 점화 제공 페트리 네트인 Marked Graph의 모델링[15]에 사용되고 있다. Marked Graph에는 각 플레이스가 단하나의 입력 트랜지션과 단하나의 출력 플레이스를 갖을 수 있다.

[정의 3.1] 페트리 네트의 모든 트랜지션들의 집합 T의 부분 집합인 T_c에 대해정의의 부분 집합 $T_c \subset \{ t_i \}$를 점화하는 것이 $\{ t_i \}$에 속하지 않는 또 다른 어떤 트랜지션 $T_c \ni t_j$를 점화하지 못하게 만든다면 T_c가 갈등구조를 이룬다고 한다.

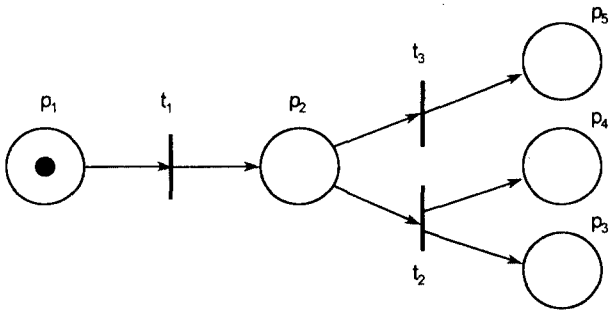


[그림 3.1] 갈등 구조의 예

그림 3.1에는 이와 같은 조건의 예를 몇 가지 보여주고 있다. 여기서 주의해야 할 것은 입력 플레이스를 공동으로 공유하는 트랜지션들이 반드시 갈등 구조를 이룰 필요는 없다. 만약에, 공유되는 입력 플레이스들에 충분한 토큰이 있어 곤란 상태에 있는 각 트랜지션들을 점화 가능하게 만들어 준다면 이들 트랜지션들은 갈등 구조를 이루지 않는다.

강한 점화 모드는 갈등 구조(conflict structure)를 갖은 특정 네트의 경우에 다음 예제에서 보여주는 것과 같이 무순에 도달하기 때문에 적합하지 않다. 그림 3.2을 보면, t_1이 점화하면 토큰이 플레이스 p_2에 추가된다. 강한 점화 모드에 따르면, 토큰이 p_2에 도착하는 동시에 각 트랜지션 t_2와 t_3가 발사 허용되어 점화를 시작해서 결과적으로 토큰이 p_3, p_4, p_6에 각각 추가된다. 이것은 갈등 상태에 있는 트랜지션들이 모두 다 점화할 수 없다는 갈등 구조의 정의에 모순이 되는 것이다.

앞에서 언급했듯이 일반 페트리 네트가 약한 점화 모드를 따르고 있고, 약한 점화 모드는 트랜지션을 반드시 점화하도록 강요하지 않기 때문에 갈등 구조가 있더라도 문제가 생기지 않는다.



[그림 3.2] 강한 점화 모드에서 갈등 구조는 모순에 도달한다

3.2 시간적인 제한의 관점

부과된 시간적 제한들은 상수 혹은 함수로 표현된다. 시간적 제한의 상수 표현에는 단 하나의 지연 시간으로 취급하는 timed Petri net[11][12], 상하 범위로 구성된 하나의 시간 쌍으로 취급하는 time Petri net[9][10]과 두 가지 둘 다 취급하는 timing constraint Petri net[3]등이 있으며, 함수 표현에는 트랜지션 점화 율의 확률 함수로 취급하는 stochastic Petri net[13]와 입력 플레이스에 있는 채색 토큰들의 함수로 취급하는 ER net[14] 등이 있다.

Timed Petri net(TPN)는 비동기적 동시발생 시스템의 시간적 조절(timing)을 조사한 Ramchandani가 처음 제안했고 Ramamoorthy와 Ho[12]에 의해 성능 평가에 이용할 수 있게 확장되었다. TPN은 강한 점화 모드를 따른다: 지연 시간 $Tdel$을 갖은 트랜지션 t_j는 필요로 하는 토큰들이 시간 $T0$에 도착했을 때 즉시 점화한다. 시간 $T0$ 전에 도착한 토큰들은 보존되지 않으며 만약에 t_j가 갈등 구조에 속해 있으면 이 구조의 다른 트랜지션의 발사 허용을 위해 이용될 수도 있다. 시간 간격 $T0$과 $\{T0+Tdel\}$사이에는 t_j의 토큰들이 보존되고, 다른 트랜지션은 사용할 수 없다. 시간 $\{T0+Tdel\}$에서 토큰들이 t_j의 입력 플레이스에서 출력 플레이스로 반드시 이동되어야 한다.

Stochastic Petri net(SPN)도 또한 성능 분석을 위해 주로 사용된다. TPN에서 상수 지연 시간을 이용한 반면에 SPN에서 트랜지션의 점화율의 확률 함수인 평균 지연 시간을 사용한다. TPN와 SPN는 주기적으로 실행하는 동안에 트랜지션이 얼마나 빨리 연속 점화를 할 수 있는가 등의 성능 평가를 위해 주로 이용한다. 다르게 이야기하면, 성능의 평가는 초기 마킹 상태에 돌아 올 수 있게 하는 점화 순서가 점화를 마치는데 소요되는 최소 주기(minimum cycle time)를 찾는 것, 즉 정기적인 프로세스의 실행 최소 주기를 찾는다.

Time Petri net(Time PN)는 Merlin과 Farber[10]가 통신 프로토콜의 회복 가능성 분석을 위해 처음 소개했다. Time PN는 TPN와 비슷하나 단 하나의 지연 시간 대신에 시간 쌍(time pair)을 쓴다. 트랜지션에는 최소 지연을 표현하는 TC_{MIN}과 최종 기한을 의미하는 TC_{MAX}으로 구성된 (TC_{MIN}, TC_{MAX})이 할당된다. 만약에 트랜지션 t_j가 시간 $T0$에 발사 허용되었다면, t_j가 ($T0+ TC_{MIN}$) 전이나 ($T0+ TC_{MAX}$) 이후에 점화할 수 없다. Time PN가 강한 점화 모드를 따르기 때문에, ($T0+ TC_{MIN}$)와 ($T0+ TC_{MAX}$) 사이에 점화하지 못하면, t_j는 ($T0+ TC_{MAX}$)에서 반드시 점화해야 한다. Leveson과 Stolzy등은 time PN를 실시간 시스템의 안전성 분석[9]에 사용했다. 안전성을 분석하기 위해 시스템의 특성을 모델링한 time PN의 도달성 그래프를 작성하여 위험 상태(risk state)에 도달할 수 있는지를 조사한다. 그 다음, 그러한 위험 상태를 피할 수 있도록 모델링한 시스템으로부터 시간적 제한들을 산출한다. Berthomieu와 Diaz는 동시발생의 시스템의 시간적(temporal) 특성을 분석하는 계산 방법을 제안했다[7]. Time PN의 도달성 분석 방법을 이용하여 그들은 모델링한 시스템의 특성을 철저하게 확인할 수 있다고 주장한다.

Timing constraint Petri net(TCPN)는 Tsai와 Yang 그리고 Chang등이 실시간 시스템에 명세서의 스케이듈 가능성 분석에 적용하기 위해 처음 소개했다. 앞에서 말한 바와 같이, TCPN는 단 하나의 지연 시간($Tdel$)과 상하 범위로써 구성된 시간 쌍 (TC_{MIN}, TC_{MAX})을 함께 취급한다. 지연 시간 $Tdel$ 은 $FIRE_{dur}$(t_j)로 표시하며 지속 시간이라고 한다. 특히, 다른 페트리 네트들과 크게 다른 점이 있다면, 플래이스에도 시간 쌍 (TC_{MIN}, TC_{MAX})을 할당했고, 트랜지션 점화 규칙이 약한 점화 모드를 사용하며, 분석 방법에 상대적(relative)인 시간 모드 외에도 절대적(absolute)인 시간 모드를 쓴다는 것이다. 만든이들은 이와 같은 구조의 가장 큰 장점으로 갈등(conflict) 구조(우선 순위, 결정, 그리고 선택 구조)를 지원하는 것을 들고 있다. 약한 점화 모드의 또 다른 장점으로서 일반 페트리 네트의 모델링 규칙을 위반하지 않고서도 사라진 토큰(lost token)을 모델링 할 수 있다는 것을 보여준다.

3.3 요약

위의 확장들을 우리는 그들의 점화 규칙에 있어서의 차이점으로 구분하여 요약한다. Timing constraint Petri net를 제외한 나머지 페트리 네트들은 모두 트랜지션이 발사 허용된 즉시 점화하도록 강요하는 강한 점화 모드를 사용한다. 트랜지션 점화는 특정 시간동안에 지속될 것이며, 점화하는 동안에는 토큰이 보존된다. Time Petri net의 경우 트랜지션의 점화가 순간적으로 이뤄지기 때문에 토큰의 보존이 안된다.

[표 1] 발사 허용 및 점화 규칙의 관점

페트리 네트	발사 허용 규칙		점화 모드		토큰 보존
	타입 있음	타입 없음	약한	강한	
Timed		*		*	Y
Time		*		*	N
Timing Constraint		*	*		Y
Stochastic		*		*	N
Colored	*			*	Y

[표 2] 시간적 제한의 관점

페트리 네트	시간적 제한				분석 시간 모드	
	상수		함수			
	단 하나의 지연 시간	시간 쌍	확률	그 외	상대적	절대적
Timed	*				*	
Time		*			*	
Timing Constraint	*	*			*	*
Stochastic	*		*		*	
Colored	*			토큰 조합	*	

4. 결 론

이 논문에서 우리는 실시간 동시발생 시스템을 설계 및 분석할 때의 시간 개념과 시간적인 제한(timing constraint)들의 역할에 대해 알아봤으며, 시간 관련 페트리 네트 확장들을 살펴봤다. 이들 페트리 네트들은 각각 자신의 방법으로 일반 페트리 네트에 시간 개념을 추가시켰다. 시간 개념이 주로 트랜지션 발사 허용 및 점화 규칙에 추가되었기 때문에 분류 기준을 발사 허용 및 점화 규칙의 관점과 시간적인 제한의 관점으로 정하였다. 조사 결과 Timing constraint Petri net가 일반 페트리 네트와 가장 비슷하면서(동일한 점화 모드를 사용) 갈등 구조(conflict structure, 실시간 동시발생 시스템에 꼭 필요하다)의 구현과 분석을 제공할 수 있기 때문에 실시간 동시발생 시스템의 설계 및 성능 분석을 위해서 제일 적합할 것으로 생각된다.

5. 참 고 문 헌

[1] Peterson,J.L., Petri Net Theory and Modeling of Systems, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[2] Cassandras,C.G., Discrete Event Systems: Modeling and Performance Analysis, Richard D.Irwin, Inc. and Aksen Associates, Inc.,1993.

[3] Tsai J.J.P., and S.J.Yang, and Y.H.Chang, "Timing Constraint Petri Nets and Their Application to Schedulability Analysis of Real-Time System Specifications",IEEE Trans. Software Eng., January 1995.

[4] Haban,D. and Shin,K.G., "Application of real-time monitoring to scheduling tasks with random execution times",IEEE Trans. Software Eng., vol. SE-16, pp. 1374-1389, Dec. 1990.

[5] Stoyenko,A.D., Hamacher,C., and Holt,R.C., "Analyzing hard real-time programs for guaranteed schedulability", IEEE Trans. Software Eng., vol. SE-17, pp. 737-750, Aug. 1991.

[6] Dasarathy,B.,"Timing constraints of real-time systems: constructs for expressing them, methods of validating them", IEEE trans. Software Eng., vol. SE-11, pp. 80-86, Jan. 1985.

[7] Berthomieu,B. and Diaz,M.,"Modeling and verification of time dependent system using time Petri nets",IEEE Trans. Software Eng.,vol.SE-17, pp. 259-273, Mar. 1991.

[8] Coolahan,J.E.,Roussopoulos,Jr.and N.,"Timing requirements for time driven systems using augmented Petri nets",IEEE Trans. Software Eng., vol.SE-9, pp. 603-616, Sept. 1983.

[9] Leveson,N.G., Stolzy,J.L.,"Safety analysis using Petri nets",IEEE trans. Software Eng., vol. SE-13,no. 3, pp. 386-397, Mar. 1987.

[10] Merlin,P.M.,Farber,D.J.,"Recoverability of communication protocols implications of a theoretical study",IEEE Trans. Commun., vol. COM-24, pp. 1036-1043, Sept. 1976.

[11] Holliday,M.A.,Vernon,M.K.,"A generalized timed Petri net model for performance analysis", IEEE Trans. Software Eng.,vol.SE-13, pp. 1297-1310, Dec.1987.

[12] Ramamoorthy,C.V.,Ho,G.S.,"Performance evaluation of asynchronous concurrent systems using Petri nets", IEEE Trans. Software Eng.,vol.SE-6, pp. 440-449, Sept.1980.

[13] Lopez-Benitez,N.,"Dependability modeling and analysis of distributed programs", IEEE Trans. Software Eng., vol. SE-20, pp. 345-352, May 1994.

[14] Felder,M., Mandrioli,D., Morzenti,A.,"Proving properties of real-time systems through logical specifications and Petri net models",IEEE Trans. Software Eng., vol. SE-20, pp. 127-141, Feb. 1994.

[15] Murata,T.,"Petri Nets: properties, analysis and application",Proc. IEEE, vol.77, pp. 541-580, Apr. 1989.